Extract Numbers From a String

See Also

Extracting numbers from text strings, removing unwanted characters , Michael Cleverly, comp.lang.tcl, 2002-06-23

An explanation with several examples.

Description

The following regular expression matches an optional leading + or -, an optional integer part, an optional decimal point, more digits, and an optional trailing exponent.

```
[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?
```

The tricky part about this expression is that in the absence of a ., the part of the pattern that normally matches the mantissa matches the integer part instead.

A similar but longer expression takes a different approach to make the the integer portion optional, adding an extra branch (|). (The original version was posted to comp.lang.tcl by Roland B. Roberts.):

```
[-+]?(?:[0-9]+(?:\.[0-9]+)?|\.[0-9]+)(?:[eE][-+]?[0-9]+)?
```

When extracting numbers from text, in order to allow separators in significant digits while avoiding picking up those separators when they occur elsewhere, a more complex expression is required:

```
# uses extended syntax
set pattern {
    # any initial + or - characters
    [-+]*
    # order of the branches matters
    (?:
            # only significant digits
            [0-9_,]*[0-9]
        |
            # only mantissa
            \.[0-9]+
        |
            # the significant digits
            [0-9_,]*[0-9]
            # the mantissa
            \.[0-9]+
    )
    # optional exponent
    (?:
        [eE^][-+]?[0-9]+
    )?
}
```

To add support for ratios, reuse the pattern:

```
set rpattern $pattern(?:\s*/\s*$pattern)?

set text "some, text.  +100 . more text. -200 h l 6.62607015e-34 1,000 xd
100,000,000.234, and 34. , 1.67262171E-27 .22"

regexp -inline -all $pattern $text; #-> +100 -200 6.62607015e-34 1,000
100,000,000.234 34 1.67262171E-27 .22
```

More information here .

WJG 2022-10-01 PYK 2022-10-09: A quick snippet on extracting a list of numbers from a string without using regular expressions:

```
proc extractNumbers str {
        set res ""
        foreach c [split $str ""] {
                if { [string is integer $c] } {
                        set a 1
                        append res $c
                } elseif { $c eq "," || $c eq "." } {
                        if {$a} { append res $c }
                } else {
                        set a 0
                        append res " "
                }
        }
        return [string trim $res]
}
```

WJG 2022-10-03 PYK 2022-10-09: Made some changes to the above procedure to allow for sub-string prefixes (+-) and infixes (.,/^). Seeing as a numeric sequence could end a clause which would append a either a comma or full-stop as sentence punctuation, these are removed from any result.

```
proc extractNumbers str {
        set buff ""
        set res ""
        set lc ""

        set pf "-+"                ;# number sequence prefixes
        set if "./ ^"        ;# number sequence infixes

        # parse the string character by character
        foreach c [split $str ""] {
                # respond to integers
                if { [string is integer $c] } {
                        set a 1        ;# toggle START of integer sequence
                        if {[string first $lc $pf] != -1 } { append buff $lc }
                        append buff $c
                } elseif { [string first $c $if] != -1 } {
                        if {$a} { append buff $c }
                } else {
                        set a 0 ;# toggle END of integer sequence
                        append buff " "
                }
                # keep tally for potential prefixes
                set lc $c
        }

        # remove sentence punction and reformat list
        foreach item $buff { lappend res [string trimright $item $pf$if] }

        return $res
}
```

in the following example, one deficiency is evident: An isolated comma or period is not properly handled:

```
extractNumbers $text; #-> +100 {} -200 6.62607015 -34 1,000 100,000,000.234 34 {}
1.67262171 -27 .22
extractNumbers "1/25 3.123^4 10^6"; #-> 1/25 3.123^4 10^6
```

WJG (13/10/22) Thanks for the comment. Not 'handling' isolated commas or periods is not a deficiency here. Both would indicate either a malformed sentence or number.